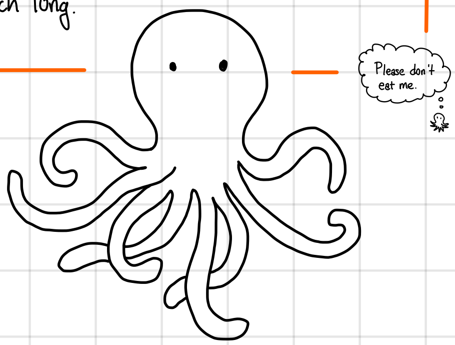
CS250/EE387 - LECTURE 9 - BACK to CONCATENATED CODES and the RANDOM CHANNEL MODEL.

TODAY'S OCTOPUS FACT

Giant Pacific Octopuses can weigh over 600 lbs and have an armspan of over 30 feet. The Wolfi Octopus generally weighs less than a gram and is under an inch long.



AGENDA

1. RANDOM CHANNEL MODEL
2. SHANNON'S THM (statement)
3. CONCATENATED CODES ACHIEVE CAPACITY (BUT...)

1 RANDOM CHANNEL MODEL

- So far, we have focused on the trade-off between **RATE** and **DISTANCE**.
- We chose **DISTANCE** because it nicely captures **WORST-CASE** error/erasure tolerance.
- Moreover, **DISTANCE** was nice for applications like compressed sensing and group testing.

- **HOWEVER**, the worst-case error model is pretty pessimistic. This motivates a **RANDOMIZED MODEL** for errors.

NOTE. The **RANDOM** (or **STOCHASTIC** or **SHANNON**) model is extremely well-studied and we will largely ignore it in this class. See EE 276 (Information Theory) or EE 388 (Modern Coding theory) for more on this very cool topic!

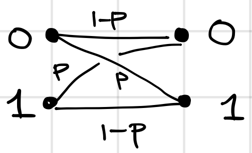
The model is this:

DEF. A MEMORYLESS CHANNEL W with input alphabet $\mathcal{X}$ and output alphabet $\mathcal{Y}$ is specified by a probability distribution,

$$W(y|x) = \text{“the probability that } y \text{ came out of } W \text{ given that } x \text{ went in.”}$$

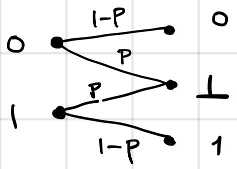
EXAMPLE: $\mathcal{X} = \mathcal{Y} = \{0,1\}$, and $W(y|x) = \begin{cases} p & y \neq x \\ 1-p & y = x \end{cases}$ for $p \in (0,1)$.

Thus, W flips a bit with probability p.
We draw this as



DEF. This channel is called the BINARY SYMMETRIC CHANNEL, BSC(p).

EXAMPLE: $\mathcal{X} = \{0,1\}$, $\mathcal{Y} = \{0,1,\perp\}$, with W given by:

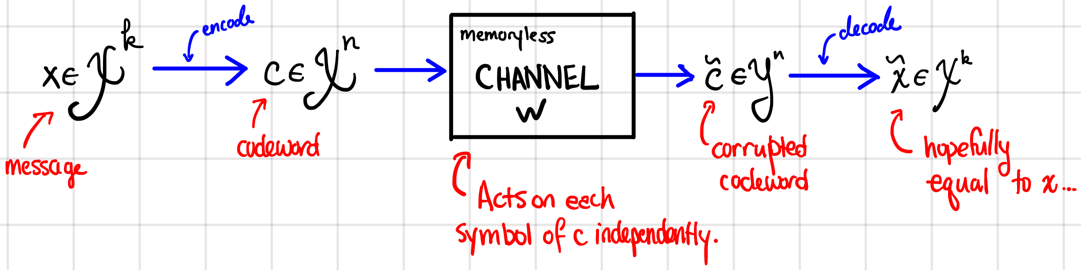


That is, W ERASES a bit with probability p.

DEF. This channel is called the BINARY ERASURE CHANNEL BEC(p).

These channels are "memoryless" because they act on one bit at a time, independently.

Our picture of error correcting codes thus looks like:


$x \in \mathcal{X}^k$
encode
$c \in \mathcal{X}^n$
memoryless
CHANNEL
W
$\hat{c} \in \mathcal{Y}^n$
decode
$\hat{x} \in \mathcal{X}^k$
message
codeword
Acts on each symbol of c independantly.
corrupted codeword
hopefully equal to x...

DEF. Let $C \subseteq \mathcal{X}^n$ be an error correcting code with encoding map $\text{ENC} : \mathcal{X}^k \to \mathcal{X}^n$ and decoding map $\text{DEC} : \mathcal{Y}^n \to \mathcal{X}^k$.

Let W be a channel w/ input alphabet $\mathcal{X}$ and output alphabet $\mathcal{Y}$.

The FAILURE PROBABILITY of C on W is at most η if $\forall\ x \in \mathcal{X}^k$,

$$\mathbb{P}_W\left(\text{DEC}\left(\underbrace{W(\text{ENC}(x))} \right) \neq x \right) \leq \eta$$

This is a random variable which represents the output of the channel W on the input $\text{ENC}(x)$.

Shannon showed that every channel has a CAPACITY, $C \in [0,1]$, so that transmitting at rate $R > C$ reliably is impossible, but transmitting at rate $R < C$ <u>is</u> possible.

This is what Shannon's Theorem says for the BSC:

THM (SHANNON'S CHANNEL CODING THM for the BSC).

$\forall\ p \in [0, ½)$ and all $\varepsilon \in (0, ½ - p)$, the following holds for large enough n:

(1) For $k \leq \left\lfloor (1 - H_2(p+\varepsilon)) \cdot n \right\rfloor$,
$\exists\ \delta > 0$, and $\text{ENC} : \{0,1\}^k \to \{0,1\}^n$, $\text{DEC} : \{0,1\}^n \to \{0,1\}^k$ s.t. $\forall\ x \in \{0,1\}^k$,

δ is independent of n

$$\mathbb{P}_{BSC_p}\left\{ \text{DEC}\left(BSC_p(\text{ENC}(x)) \right) \neq x \right\} \leq 2^{-\delta n}$$

aka, if the rate is a smidge below $1 - H(p)$, the failure prob. can be really tiny.

(2) If $k \geq \left\lceil (1 - H_2(p) + \varepsilon) \cdot n \right\rceil$, then for all such ENC, DEC,

$$\mathbb{P}_{BSC_p}\left\{ \text{DEC}\left(BSC_p(\text{ENC}(x)) \right) \neq x \right\} \geq ½ .$$

aka, if the rate is a smidge above $1 - H(p)$, the failure prob. is at least ½.

ASIDE. More generally, Shannon's Thm says that the capacity of W is $C = \underset{\text{over dists on inputs } X}{\max} \overset{\text{mutual information}}{I}(\underset{\text{what goes in to the channel}}{X};\underset{\text{what comes out}}{Y})$

The proof of Shannon's theorem is best done w/ information theory (see EE 276).

Here's a handwavy sketch of an argument to prove the BSC case directly:

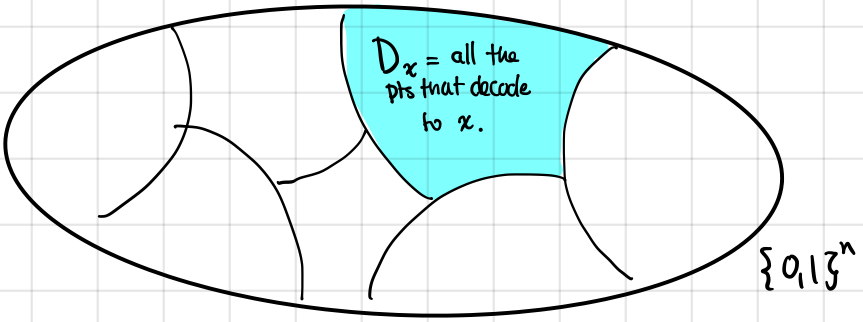
pf (sketch)

(1) A random code works great* for the achievability result.

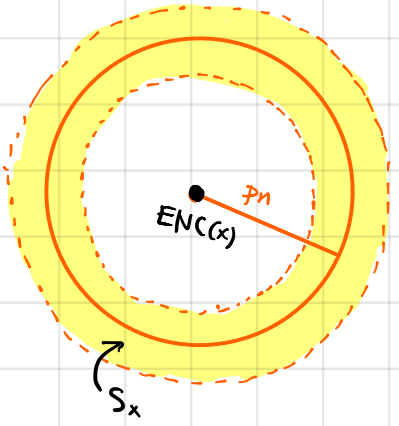
* actually, we'll have to modify the random code a bit by throwing out a few bad codewords.

(2) For the impossibility result, consider dividing up $\{0,1\}^n$ into a bunch of chunks,

$$D_x = \{ y \in \{0,1\}^n \mid DEC(y) = x \}.$$

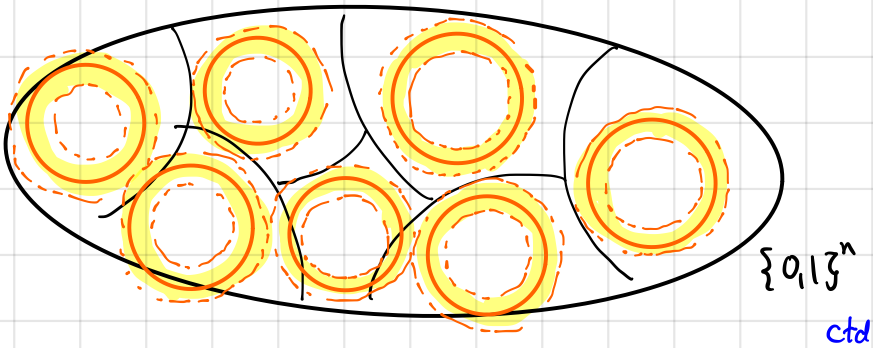


Now, consider what happens to ENC(x) when it goes through the BSC:



The corrupted version is REALLY likely to end up in this annulus, S_x.

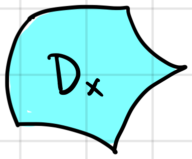
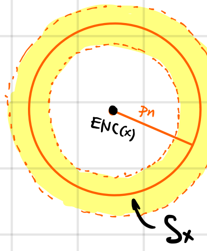
Thus, we better have that most of S_x is contained in D_x, or there would be some big probability of failure.



ctd.

pf sketch ctd.

But then we should have, for all x:

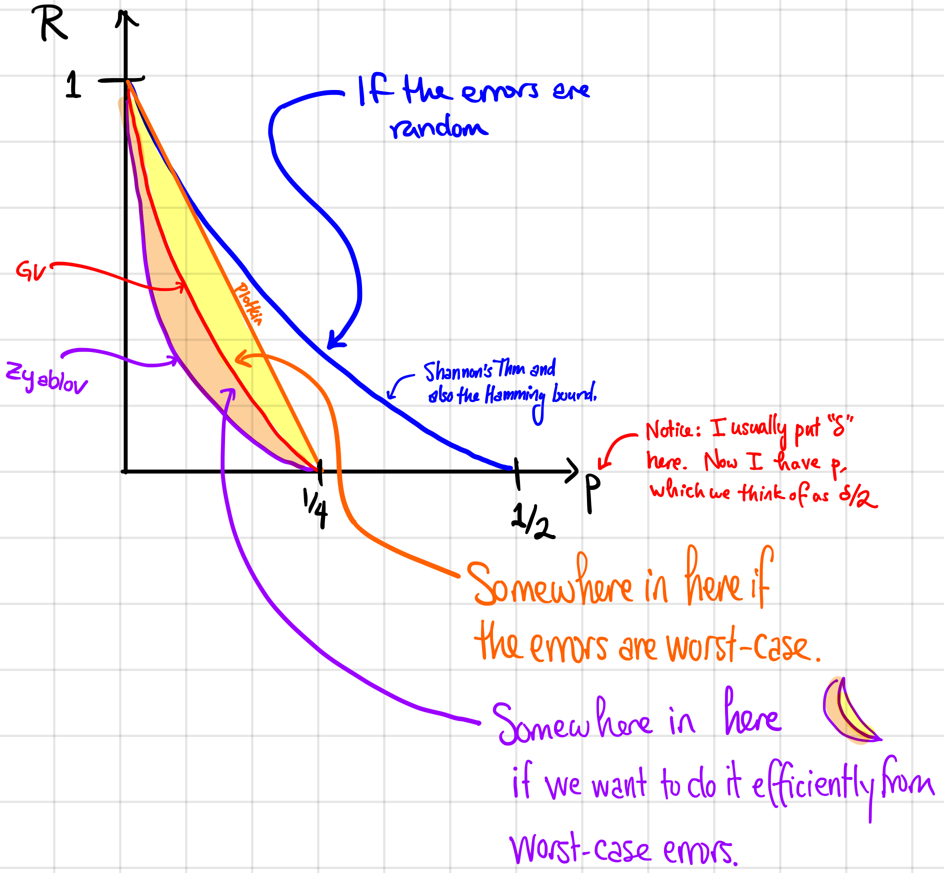
Vol () $\gtrsim$ Vol () $\approx \mathrm{Vol}_2(pn,n) \approx 2^{n\cdot H(p)}$

"$\gtrsim$" here means "biggerish than"

So then how many D_x's could possibly fit in $\{0,1\}^n$? At most $2^n / 2^{n\cdot H(p)} = 2^{n(1-H(p))}$

So $|C| \lesssim 2^{n(1-H(p))}$, so $R \lesssim 1-H(p)$ [or so].

So, to recover from a p-fraction of errors:



If the errors are random

Notice: I usually put "δ" here. Now I have p, which we think of as $\delta/2$

Somewhere in here if the errors are worst-case.

Somewhere in here if we want to do it efficiently from worst-case errors.

Natural question: What if I want to efficiently decode from random errors?

(3) CONCATENATED CODES achieve CAPACITY

THM. For every p and every $\varepsilon \in (0, 1-H_2(p))$, and all large enough n, there is a binary linear code $\mathcal{C} \subseteq \{0,1\}^n$ with rate $R \geq 1-H_2(p)-\varepsilon$, so that:

(a) $\mathcal{C}$ can be constructed in time $\mathrm{poly}(n) + 2^{O(1/\varepsilon^5)}$

(b) $\mathcal{C}$ can be encoded in time $O(n^2)$

(c) There is a decoding alg DEC for $\mathcal{C}$ that runs in time

$$\mathrm{poly}(n) + n \cdot 2^{O(1/\varepsilon^3)}$$

and has failure probability at most $2^{-\Omega(\varepsilon^6 n)}$ over $BSC(p)$.

Thus, this code "acheives capacity" on the BSC, in the sense that the rate can get arbitrarily close to $1-H_2(p)$.

DRAWBACK! As the rate gets close to $1-H_2(p)$, the running time of these algorithms blow up EXPONTIALLY in $1/\varepsilon$.

Whether or not this could be avoided (with efficient algs) was open for a long time... but then in 2009 Arikan introduced **POLAR CODES** which will do it. We might talk about polar codes later in class. ← And if not, it's a great project topic!

But for now let's prove (or, sketch the proof of) this theorem.

It turns out, we've already seen the answer! Concatenated codes!

PROOF SKETCH for the THEOREM:

Choose a parameter γ TBD.

CODE CONSTRUCTION: Concatenated Code with:

Code	Dimension	Block len.	$\lvert\Sigma\rvert$	Rate	Decoding Time	Other
C_{in}	k_{in}	n_{in}	2	$1-H_2(p)-\varepsilon/2$	$T_{in}(n_{in})$	Fail prob on BSC$_p$: $\gamma/2$
C_{out}	k_{out}	n_{out}	$2^{k_{in}}$	$1-\varepsilon/2$	$T_{out}(n_{out})$	Distance: 2γ

↖ Both C_{in} and C_{out} will be linear

We'll see how to get these in a moment...

RATE is $R = (1-H_2(p)-\varepsilon/2)\cdot(1-\varepsilon/2) \geq 1-H_2(p)-\varepsilon.$ ✓

DECODING ALG is the one that wasn't a good idea last week:

```
Given $(y_1,\dots,y_{n_{out}}) \in (\mathbb{F}_2^{n_{in}})^{n_{out}}$
FOR $i=1,\dots,n_{out}$:
    Use $C_{in}$'s decoder to obtain $y_i' = \mathrm{DEC}_{in}(y_i) \in \mathbb{F}_2^{k_{in}} \cong \mathbb{F}_{q_{out}}$
Decode $y' = (y_1',\dots,y_{n_{out}}')$ using $\mathrm{DEC}_{out}$ ($C_{out}$'s decoder), and
RETURN $(\mathrm{DEC}_{out}(y'))$
```

Say DEC_{in} takes time $T_{in}(n)$, DEC_{out} takes time $T_{out}(n)$.
Then the decoding time is

$$\text{DECODING TIME} = O\left(n_{out}\cdot T_{in}(k_{in}) + T_{out}(n_{out}) \right) = \text{TBD}$$

ENCODING TIME $= O(n^2)$, since the code is linear

CONSTRUCTION TIME: TBD

ERROR PROBABILITY:

Say that C_{out} has relative distance γ. Then

$$\mathbb{P}\{\text{decoder fails}\} = \mathbb{P}\{ > \gamma \cdot n \text{ blocks are incorrectly decoded by } C_{in}\}$$

For a fixed block i, $\mathbb{P}\{ C_{in} \text{ decodes the } i^{th} \text{ block incorrectly}\} \leq \gamma/2$.

Each bit is flipped independantly, so

$$\mathrm{BSC}(\square\square\square\square) \sim \mathrm{BSC}(\square)\,\mathrm{BSC}(\square)\,\mathrm{BSC}(\square)\,\mathrm{BSC}(\square)$$

So $\mathbb{P}\{ \geq \gamma \text{ blocks are in error}\}$

$$\leq \mathbb{P}\left\{ \frac{1}{n_{out}} \sum_{i=1}^{n_{out}} \mathbb{1}\left\{ \begin{array}{l} C_{in} \text{ fails} \\ \text{on block } i \end{array} \right\} > \gamma \right\}$$

$$\leq \mathbb{P}\left\{ \frac{1}{n_{out}} \sum_{i=1}^{n_{out}} \mathbb{1}\left\{ \begin{array}{l} C_{in} \text{ fails} \\ \text{on block } i \end{array} \right\} > 2 \cdot \mathbb{E}\left[\frac{1}{n_{out}} \sum_{i=1}^{n_{out}} \mathbb{1}\left\{ \begin{array}{l} C_{in} \text{ fails} \\ \text{on block } i \end{array} \right\} \right] \right\}$$

This follows from a "CHERNOFF BOUND."

$$\leq \exp(-\gamma \cdot n_{out} / 6)$$

So the error probability is indeed exponentially small.

FUN EXERCISE: A random linear code of rate $1-H(p)-\epsilon/2$ (probably) has fail prob. $2^{-\Omega(\epsilon^2 n)}$.

So choose $k_{in} = \Omega\left(\frac{\lg(1/\gamma)}{\epsilon^2}\right)$ and we know there EXISTS a binary linear code that works.

But now.... What codes to use for C_{in}, C_{out} ??

INNER CODE: Just like before, let's try ALL the binary linear codes.

CONSTRUCTION TIME: $2^{O(n_{in}^2)}$: there are $\leq 2^{k_{in} \cdot n_{in}}$ codes to check, and it takes time $2^{k_{in}} \cdot 2^{O(n_{in})}$ to compute the error probability for each one.

#codewords c that might be transmitted

time to compute $\sum_{y \in \{0,1\}^{n_{in}}}' \mathbb{P}\{y|c\} \cdot \mathbb{1}\{\mathrm{DEC}_{IN}(y) \neq c\}$

DECODING TIME: $2^{O(k_{in})}$ to try all the codewords and find the closest one.

OUTER CODE:

TRY 1: REED-SOLOMON. Actually, NO! Like we saw last week, this would require $n_{out} = 2^{k_{in}}$ but then the construction time would be $2^{O(k_{in}^2)} = n_{out}^{\log(n_{out})}$, and we get a quasipolynomial-time construction.

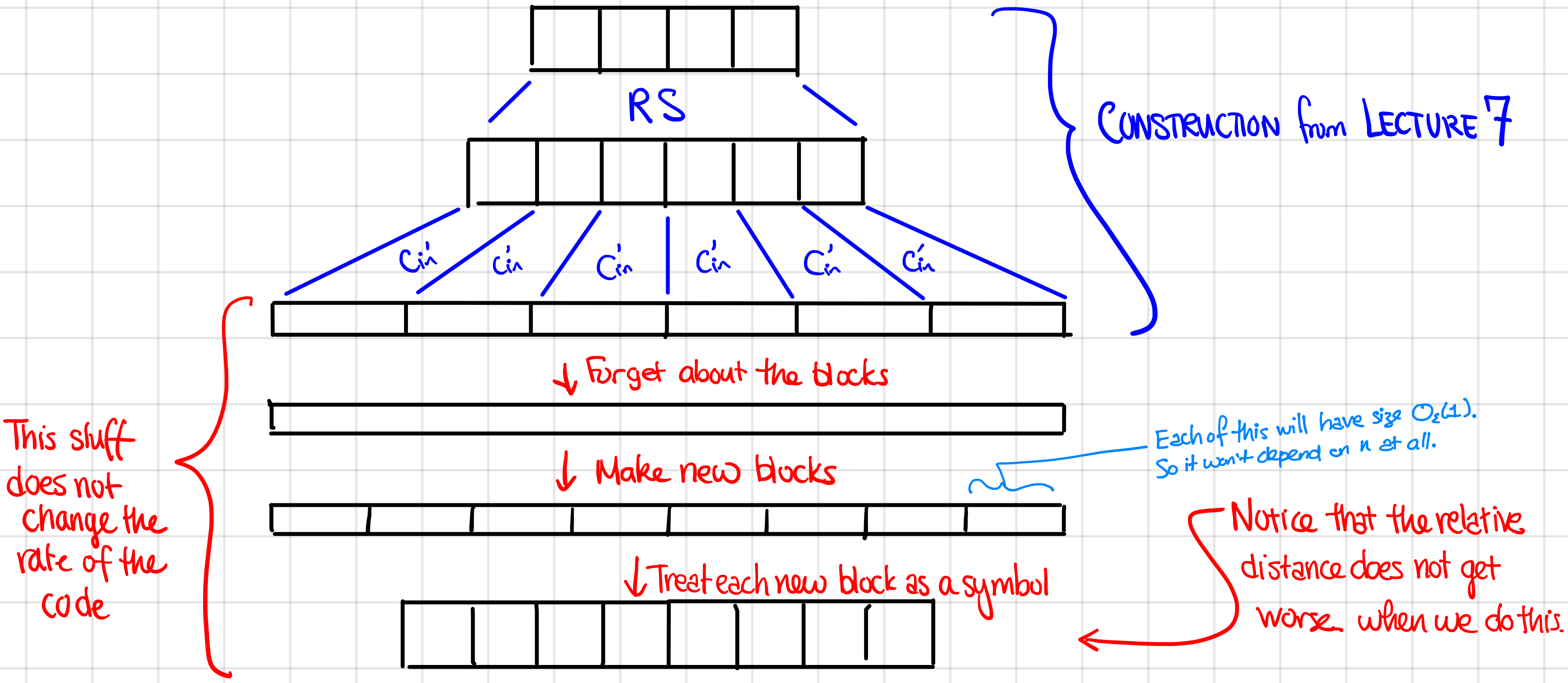
Before, we got around this by coming up with a slightly better construction of C_{in}, that took time $2^{O(k_{in})}$ instead of $2^{O(k_{in}^2)}$. Here, we'll mess with the outer code instead.

TRY 2: BINARY CODES on the ZYABLOV BOUND.

Meta-logic: We need an efficiently encodable/decodable asymptotically good code. So far, we have seen:

- RS CODES ← We just said NOT this.
- Concatenated RS CODES ← Better be this.

So let C_{out} be an explicit binary code on the Zyablov bound.


RS
C_{in}'
C_{in}'
C_{in}'
C_{in}'
C_{in}'
C_{in}'
CONSTRUCTION from LECTURE 7
Forget about the blocks
Make new blocks
Each of this will have size $O_\varepsilon(1)$. So it won't depend on n at all.
Treat each new block as a symbol
This stuff does not change the rate of the code
Notice that the relative distance does not get worse when we do this.

Since the rate and distance don't get worse, this code STILL lies at or above the Zyablov bd. To decode, just run the red stuff backwards and then use the decoder from Lec. 7.

PARAMETERS:

- Choose C'_{out} to be a binary code on the Zyablov bd (from Lecture 7).
- Choose $k_{in} = \Theta\left(\frac{\log(1/\gamma)}{\epsilon^2}\right)$ ← This is what we needed for the inner code to exist.
- Make $C_{out} \subseteq \left[\mathbb{F}_{2^{k_{in}}}\right]^{n_{out}}$ by chopping up C_{out} into chunks of size k_{in}.
- Now let's pick γ. We have

$$\delta_{out} = (1-\underbrace{R_{RS}}_{\text{Rate of the RS code on prev. page}})\,H^{-1}(1-\underbrace{r}_{\text{this was the rate of } C_{in}' \text{ on the previous page — NOT } C_{in} \text{ in our construction}}) \quad \leftarrow \text{Zyablov Bd}$$

and recall we want $\delta_{out} = 2\gamma$.

So choose $R_{RS} = 1-2\sqrt{\gamma}$ and r s.t. $\underbrace{H^{-1}(1-r) = \sqrt{\gamma}}_{\text{This means } r = 1-O(\sqrt{\gamma}\lg(1/\gamma))}$,

and that implies

$$R_{out} = R_{RS}\cdot r = (1-2\sqrt{\gamma})(1-O(\sqrt{\gamma}\,\lg(1/\gamma)))$$
$$= 1-O(\sqrt{\gamma}\lg(1/\gamma)).$$

We wanted $R_{out} \geq 1-\epsilon/2$, which means that we should choose γ s.t. $\epsilon/2 = O(\sqrt{\gamma}\lg(1/\gamma))$.

$\gamma = O(\varepsilon^3)$ works, so let's do that.

With our choice of $\gamma = \varepsilon^3$, let's go back and compute stuff.

Code	Dimension	Block len.	$\lvert\Sigma\rvert$	Rate	Decoding Time	Other
C_{in}	$k_{in} = \Theta\left(\frac{\lg(1/\gamma)}{\varepsilon^2}\right) = \Theta(\varepsilon^{-2}\lg(1/\varepsilon))$	$n_{in} = \Theta(k_{in})$	2	$1 - H_2(p) - \varepsilon/2$	$T_{in}(n_{in}) = 2^{O(k_{in})} = 2^{O\left(\frac{\lg(1/\varepsilon)}{\varepsilon^2}\right)}$	Fail prob on BSC_p: $\gamma/2$
C_{out}	k_{out}	$n_{out} = \frac{n}{n_{in}} = \Theta\left(\frac{\varepsilon^2 n}{\lg(1/\varepsilon)}\right)$	$2^{k_{in}}$	$1 - \varepsilon/2$	$T_{n_{out}}(n_z) = \text{poly}(n_{out})$	Distance: 2γ

So:

$$\text{DECODING TIME} = O\Big(n_{out} \cdot T_{in}(k_{in}) + T_{out}(n_{out})\Big) = \text{poly}(n) + n \cdot 2^{O(\lg(1/\varepsilon)/\varepsilon^2)}$$

$$\text{FAILURE PROBABILITY: } \exp\left(-\gamma \cdot n_{out}/6\right) = \exp\left(-\Omega\left(\frac{\gamma \cdot \varepsilon^2 n}{\lg(1/\varepsilon)}\right)\right) = \exp(-\Omega(\varepsilon^5 \cdot n)).$$

$$\text{CONSTRUCTION TIME: } 2^{O(n_{in}^2)} + \text{poly}(n_{out}) = 2^{O(\varepsilon^{-4}\lg^2(1/\varepsilon))} + \text{poly}(n)$$
$$= 2^{O(1/\varepsilon^5)} + \text{poly}(n).$$

and this gives all the things we claimed.

QUESTIONS to PONDER:

1. Which model (Shannon or Hamming) do you find more compelling?
2. Flesh out the details of our proof of Shannon's Thm for the BSC.
3. Why do we ask for failure probability $2^{-\Omega(n)}$? Is $1/n^{10000}$ okay?
4. Can you make the (something) $\circ$ RS approach work for acheiving capacity on the BSC?